January 1993

# Control of Visually Guided Behaviors

Jana Košecká
*University of Pennsylvania*

Ruzena Bajcsy
*University of Pennsylvania*

Max L. Mintz
*University of Pennsylvania*, mintz@cis.upenn.edu



University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-101.

# Control of Visually Guided Behaviors

## Abstract

We propose an approach for modeling visually guided behaviors of agents which explore and navigate in unknown and partially known environments. Behaviors are modeled as finite state machines (FSM), where the states of the model correspond to particular continuous control strategies and the transitions between them are caused by events representing qualitative or asynchronous changes in the behavior evolution. In order to prevent conflicts in parallel execution of multiple behaviors we adopt the supervisory control theory of discrete Event System (DES). Modeling the participating processes using the DES framework allows us to capture often complex interactions between components of the system and synthesize the resulting supervisor, guaranteeing the overall controllability of the system at the discrete event level. In the real world agents have multiple options/paths for carrying out their task. Hence there is a need for selecting different control strategies based on efficiency and safety criteria. We have included in our formalism a measure of efficiency as the nominal cost (in our case, the traversal time) and a measure of safety at the risk cost (in our case, the inverse of the distance between the agent and obstacles). Experiments have been carried out testing the described formalism with one agent carrying out the task of avoiding an obstacle in its path while tracking a target.

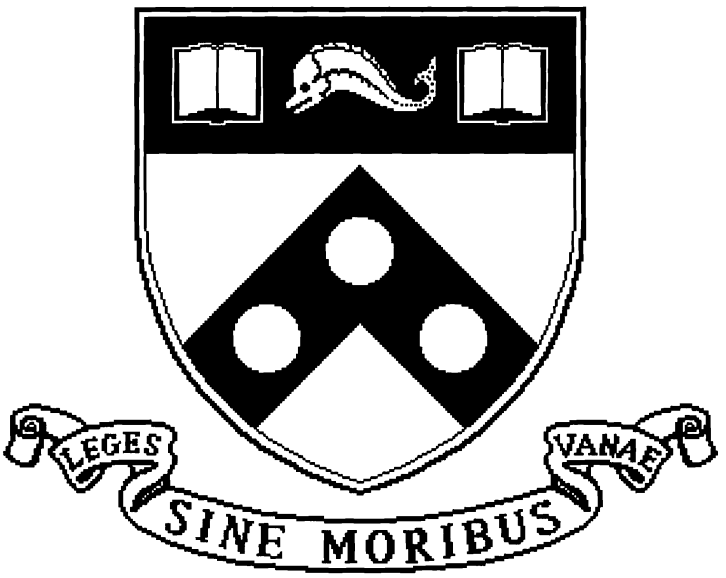
## Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-101.

# control of Visually Guided Behaviors

MS-CIS-93-101
GRASP LAB 367

Jana Košecká
Ruzena Bajcsy
Max Mintz




University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

December 1993

# Control of Visually Guided Behaviors

Jana Košecká Ruzena Bajcsy Max Mintz

GRASP Laboratory

Department of Computer and Information Science

University of Pennsylvania

Philadelphia, PA USA

## Abstract

We propose an approach for modelling visually guided behaviors of agents which explore and navigate in unknown and partially known environments. Behaviors are modelled as finite state machines (FSM), where the states of the model correspond to particular continuous control strategies and the transitions between them are caused by events representing qualitative or asynchronous changes in the behavior evolution. In order to prevent conflicts in parallel execution of multiple behaviors we adopt the supervisory control theory of Discrete Event System (DES). Modelling the participating processes using the DES framework allows us to capture often complex interactions between components of the system and synthesize the resulting supervisor, guaranteeing the overall controllability of the system at the discrete event level. In the real world agents have multiple options/paths for carrying out their task. Hence there is a need for selecting different control strategies based on efficiency and safety criteria. We have included in our formalism a measure of efficiency as the nominal cost (in our case, the traversal time) and

a measure of safety as the risk cost (in our case, the inverse of the distance between the agent and obstacles). Experiments have been carried out testing the described formalism with one agent carrying out the task of avoiding an obstacle in its path while tracking a target.

# 1 Introduction and motivation

This paper aims at the systematic analysis and design of behaviors of artificial agents. Later we shall concentrate on a specific class of cooperative behaviors. What is a behavior? A general definition is a connection of perception to action. We shall first justify why are we considering behaviors of agents rather than just perception/action systems. The issue here is what is a primitive or elementary unit, i.e. a building block which is useful and desirable for designing an intelligent autonomous system.

For last 20 years or so the Artificial Intelligencia has decoupled primitives based on perceptual signal processing, modelling, planning and execution modules. The organization of these modules, how they were connected together, was determined by the "architecture" of the system where the flow of information was strictly horizontal (the output of one module became an input for the following one). This traditional approach was criticized R. Brooks, who proposed a different organization and modularization of the system in his subsumption architecture [Bro86]. In his approach, he selected the primitives or modules along the lines of behaviors rather than along the special processing functionalities. The primitives in Brooks' design are simple reflexive behaviors, very much like those of insects: avoiding danger, attraction to light, and the like. The main idea was that higher-level behaviors subsume (hence the subsumption architecture) the lower-level behaviors, yet when for some reason the performance of the higher-level behavior fails the system does not stop but continues to function with the lower/simpler behaviors.

The criticism of Brook's proposal was that it is difficult to generalize.

In our work, we have accepted the principle that behavior is a primitive of the system. Further, we attempt to formalize this concept so that a coherent theory of complex behaviors evolves. By definition, an entity is a primitive either when it cannot be further subdivided or it is undesirable to do so. Hence, we shall have primitive behaviors which will connect simple perceptual events to appropriate actions (e.g., obstacle avoidance and tracking behaviors). Perceptual events can arise from different sensors, such as visual, contact or other noncontact sensors; actions can be mobile maneuvering, or physical or information-based manipulation (sending or receiving a message being an example of the latter). As a formalism for modeling behaviors we have selected the Discrete Event System (DES) [RW89] described in Section 3. We have identified two open problems with this behavior-based approach:

- The perception/action connection is more complex when visual sensors are used.
- There is a lack of methodology for Building composite behaviors from primitive ones while taking into account uncertainty introduced by the system and the environment.

## 1.1 Visual perception/action connection

The issue here is that visual perception is far more complex than any other modality; hence its connection to action is nontrivial. The need for integration of different visual modules, utilizing different cues at various levels, has been realized for quite some time. One type of integration stems from efforts to obtain a 3D description of the world and proposes how different vision modules should contribute to the description [AS89]. For the domain of mobile agents, where vision serves to accomplish a particular action-task,

the purposive solution turns out to be appropriate[1]. The purposive approach views vision as a collection of dedicated vision processes and focuses on extraction of qualitative information needed to accomplish the task. Even though this approach relates the task and the perceptual capabilities needed to accomplish it, the issues of control have thus far been overlooked. The early ties between perception and action were established in the area of active vision, addressing primarily control of intrinsic and extrinsic camera parameters [Bal91, CB91, FHR+90].

In more complex tasks involving navigation control has been successfully accomplished using various behavior-based methods [Bro86, Ark87]. The success of reactive behaviors was mostly due to the fact that the coupling between sensors and actuators was very tight and the sampling rate was very high. This was possible because the sensors used to demonstrate the approach were very simple (single infrared, ultrasound); therefore there was no need for selecting a particular data acquisition strategy. The combination of sensory readings and generation of the commands to the actuators was fairly straightforward. This is not the case for visual sensory data. Due to the large amount of information inherent in the visual data, we need to select acquisition and processing strategies to obtain the qualitative and quantitative information needed to control the actuators of a given system. One attempt to follow the 'classical behavioralism' can be found in [Hor93], where the constraints of the environment (ground plane, color of the carpet, etc.) were used to extract some primitives from the images which were directly coupled to the actuators. The agent successfully moved around and was able to track arbitrary moving objects (visitors) upon request.

In the long run, we would like to address more complicated behaviors/tasks, such as occur in the cooperation between two or more mobile agents, following

[1]Extensive discussion of this topic can be found in [Tea93]

one another or following a given path while avoiding unexpected obstacles. Here, the qualitative information and the choice of the control strategy are more task dependent and are affected by some risk/cost function measures which affect the behavior of the agent. We will argue that the idea of having multiple parallel perception/action processes is feasible, but in the case of systems with multiple degrees of freedom and a larger variety of tasks, there is a need for supervisory process which will guarantee the constraints imposed by the task.

In this paper we concentrate on the description of the architecture of a single mobile agent with multiple degrees of freedom whose behavior is modelled as a composition of multiple motor and perceptual processes running in parallel. We will demonstrate our approach by describing in detail the interactions between obstacle avoidance, tracking and path following behaviors. The main contribution of this paper is a systematic framework for modelling behaviors of autonomous agents and their interactions. We propose how to combine different components of the system modelled by continuous control theory techniques or reactive behaviors[2] while taking into account asynchronous interactions with the environment or other system components. Specifically, in Section 3 we introduce the notion of a **hybrid system**,[3] outline its DES model, followed with an overview of Supervisory Control Theory of Discrete Event Systems [RW89]. The description of the experimental platform used to justify our approach and experimental results is given in Section 4. There, we also present the DES models of motor and perceptual processes, address the control issues at the discrete event level and design a supervisor for the overall system. In Section 5 we conclude with a discussion of possible extensions of

[2]Reactive behaviors can be viewed as "heuristic" control theory techniques where the control law is derived experimentally without having an explicit model of the plant.

[3]The definition of the hybrid system is slightly modified from the one introduced in [ALS93]

the system.

# 2 Why DES?

Intelligent systems are typically too complex to be able to describe them by one behavior. Hence, the question we wish to answer is: what are the combination rules of more than one behavior, either within a single agent or amongst several independent agents?

The extraction of appropriate qualitative information from sensory data allows us to develop some simple obstacle detection/avoidance or target detection/tracking strategies. Most of these control strategies control the actuators in the *continuous mode*, but this mode may change to *point-to-point* or *reactive control* as a response to external environmental stimuli or the task at hand. The behaviors activated in order to achieve a given task may have conflicting effects on the actuators of the system. In the case of navigation one possible solution is to compute a "blend" (weighted average) of all preferable steering directions suggested by participating processes [FGS+92] and choose the resulting command by optimizing the blend. This approach works well when both the number of participating processes and the degrees of freedom of the system are small. As soon as the number of degrees of freedom increases or their coupling is task dependent this approach may no longer be suitable. Further, this solution requires the assumption that control is uniformly continuous. However, environmental stimuli are not necessarily continuous. As the stimuli change they may require different control for carrying out the action. This fact necessitates a consideration of discrete events and states.

In order to achieve a compact description of a hybrid system we have chosen the theory of Discrete Event Systems developed by Ramadge and Wonham [RW89]. The DES formalism models systems in terms of finite state machines (FSM). In our domain the observations — the qualitative

information extracted from sensory and encoder data — are represented in terms of **events**. With the **states** we associate particular observation/control strategies. The DES framework is suitable for investigating control-theoretic properties of the system such as **controllability** and **observability**, which can be conveniently predicted. Moreover, various visually guided behaviors can be combined in a modular and hierarchical fashion such that the resulting behavior will be *guaranteed* to be controllable.

In the following section we will introduce a more formal notion of a hybrid system and provide a framework for obtaining a DES model of a system in which behavior is governed by difference/differential equations.

# 3 Hybrid systems and the DES framework

The traditional definition of the hybrid system [ALS93] is a combination of a continuous plant and the discrete state system controller. The plant is typically modelled by difference/differential equations. The controller and the plant communicate through an interface where the plant translates the plant output $z$ to event $\tilde{z}$ (the output signal is represented by a piecewise-continuous vector-valued variable). The controller, in contrast to the model proposed by [ALS93], outputs continuous command signals $r$ for the plant input (see Figure 1). The controller is essentially a finite state machine which has a particular control law associated with each state, where the input and the state of the system are related in a following way:

$$\dot{x} = f(r, x)$$

$$z = g(x)$$

where $x \in \Re^n$, $r \in \Re^m$ and $z \in \Re^p$ are state, input and output vectors respectively. $f : \Re^n \times \Re^m \rightarrow \Re^n$ and $g : \Re^n \rightarrow \Re^p$.

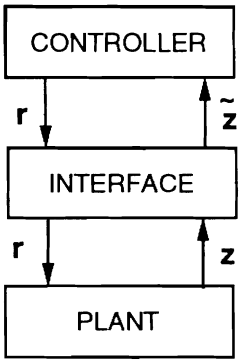



Figure 1: Model of a Hybrid System

In our case the output variables are the same as state variables, i.e. $z = x$. In order to model the plant and the controller as a discrete event system we have to identify a set of operating regions of the continuous state space and control laws which are applicable within a region. Subsequently, these regions will form equivalence classes by means of a mapping $\alpha$:

$$\tilde{z} = \alpha(x)$$

which maps the continuous observations to the set of events. An event is generated by the interface (Figure 1) when the plant state $x$ crosses the boundary between two regions of the state space.[4] This mapping abstracts the behavior of the plant and the controller to the discrete event level. In our case only the observations are discrete, while preserving the control signal continuously. Later, we will refer to this hybrid model of plant and controller as the DES plant.

Let $\Sigma$ denote the set of events. Then event trajectories can be thought of as strings over this fixed alphabet $\Sigma$. Let the subset $L \subseteq \Sigma^*$ represent all event trajectories which are physically possible for the system and fully

[4]There is no general recipe for deriving the mapping $\alpha$. Mapping reflects the granularity of description we want to achieve at the discrete event level and depends on the task we want to accomplish.

characterize its behavior. In the case when **language** $L$ is regular there exists some finite automaton $\mathcal{G}$ , such that $L$ is generated/accepted by $\mathcal{G}$ .

Let this automaton $\mathcal{G}$ be a 5-tuple

$$\mathcal{G} = (Q, \Sigma, \delta, q_0, Q_m)$$

where

$Q$ - is the set of all possible states,

$\Sigma$ - is the set of all possible events

$\delta$ - is the transition function $\delta : \Sigma \times Q \rightarrow Q$

$q_0$ - is the initial state,

$Q_m$ - is the subset of states called marker states, $Q_m \subset Q$

In order to adjoin the means of control to the system at the discrete event level, events are classified into two categories: **uncontrollable events** $\Sigma_u$ which can be observed but cannot be prevented from occurring (e.g. obstacle_detected, target_detected), and **controllable events** $\Sigma_c$ that can be prevented from occurring or forced to occur (e.g. path_computed), where $\Sigma = \Sigma_u \cup \Sigma_c$. For the types of events caused by asynchronous interaction with the environment or other processes, the proper labeling can be determined directly. Let us recall the type of event that corresponds to a crossing of a particular region boundary in the state space of the system, where with each region of the state space we have associated some control law. Here the notion of controllability is slightly modified. The event is said to be controllable when the control law associated with the region of the state space is applicable to the new state. This essentially means that if two neighboring regions have the same control law associated with them (abstracted to the one discrete state of the DES model) the events corresponding to crossing boundaries between them are controllable. This type of controllable event

originates and ends in the same state of the DES model of the plant. In the case when the control laws are different, an event is said to be uncontrollable and causes a transition to a different state. Uncontrollable events correspond to the abrupt changes in the observations of the controller where in order to preserve continuity of the output we need to change the control law (i.e. the state of DES model). This will be clearly demonstrated in the examples in the system description section. The controllable events will be denoted with ':c' throughout the figures.

## 3.1 Supervisory control

A supervisor can be thought of as a state machine in which each state has associated some control pattern determining which controllable events are enabled and which are disabled. The existence of a supervisor for a given DES plant[5], i.e. the existence of an appropriate feedback control, is closely related to the concept of **controllability**. A system is said to be **controllable** if given any initial state, there is a control law by which we can reach any desired state of the system. If the desired behavior of the plant is controllable the existence of a supervisor is guaranteed [RW87]. The control issues addressed by the DES framework differ from those in classical continuous control theory. The control at the discrete event level models the changes between different strategies triggered by abrupt observations (events) or driven by different tasks. The overall behavior of the system can be changed by changing the supervisor.

[5]The DES plant model is the model of the plant and the controller as introduced in the previous section.

## 3.2 A risk-driven cost model for cooperative behavior

### 3.2.1 A decision model

So far we have considered dynamic systems which have no uncertainty, and the criteria for discretization of events and the subsequent switching of their control strategies is rigid. In designing and operating purposeful realistic dynamic systems it is necessary to select control strategies which achieve the stated operational goals with proper regard to both efficiency and safety. The problem of control strategy selection can be modelled by a family of dynamic decision problems. These models must include adequate representations of the underlying uncertainties in the dynamic systems, sensors, and the environments. The necessity to contend with dynamic system, sensor, and environmental uncertainties is what makes this problem challenging.

There are two basic costs associated with the operation of these dynamic systems in the presence of uncertainty: (i) the nominal cost of a feasible solution based on its efficiency, e.g., time requirements and/or energy requirements; and (ii) the risk cost associated with failure modes, e.g., the cost of a collision between a mobile robot and an obstacle.

A proper system design must account for both types of costs since both efficiency and safety are important design criteria. Here, we do not necessarily require that the resulting control selection be globally optimal but merely that it suffice in that it meets the stated design objectives.

We consider two approaches to formulating these decision problems. In the first (P1), we set, a priori, a maximum allowed value for the risk cost and then minimize the nominal cost (or seek a near-minimum) subject to the risk-cost constraint. In the second (P2), we set, a priori, a maximum allowed value for the nominal cost and then minimize the risk cost (or seek a near-minimum) subject to the nominal-cost constraint. These classes of constrained optimization problems are frequently encountered in other areas,

for example, in two-class hypothesis testing, where the probability of a type I error is constrained to be no more than a given value, and the probability of a type II error is minimized subject to this constraint. The selection of P1 versus P2 depends on the values the user places on constraining each of the two costs.

# 4 System description

In this section we describe the experimental platform used to demonstrate our approach. The mobile agent in our experiments consists of a camera pan platform mounted on a TRC Labmate mobile base with two independent driving wheels. The center of rotation of the camera pan platform (camera in center in Figure 2 and the mobile base are identical. An additional stereo pair of cameras having fixed tilt angle with respect to the horizontal plane is positioned on the sides of the mobile base (see Figure 2). Each physical component of this system is characterized by its actuators, sensors and encoders. Adopting the control terminology, each such component or their composition form a plant — the subject of our control.

In order to achieve higher autonomy and accomplish more complicated tasks the notion of the plant is extended to include additional sensing capabilities. We will concentrate on the visual sensing and its specifics, but our approach can be applied to other types of sensors. For the components at hand we propose a systematic way to partition the state space, choose the set of events and subsequently determine the DES model of the given component. The system is described in more detail below.

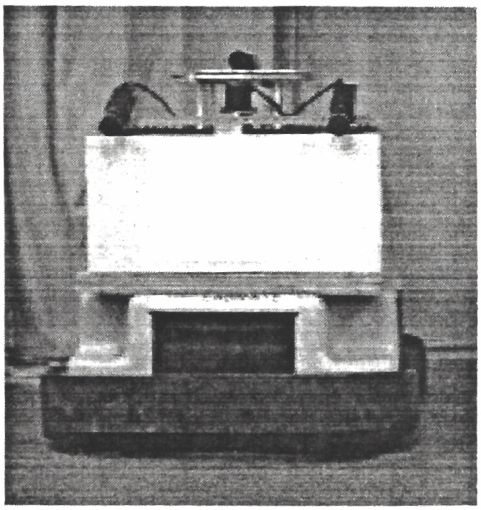


Figure 2: Mobile agent

## 4.1 Pan platform and target tracking

The platform has a dual stepper motor system which operates in two possible movement modes: *continuous* motion where the platform moves at a set velocity or *step count* motion (point-to-point motion) where the stepper motor uses the step count to move to a given position. The state of the system is described at each instance of time by $(\omega, \dot{\omega})$, where $\omega$ is a current orientation of the pan platform while moving at a given velocity $\dot{\omega}$. The camera mounted on the pan platform is used for tracking a target. The choice of suitable features to track and thereby uniquely describe the motion of the target has been addressed by several researchers [BSM+89, HA91]. For the time being we track an easily detectable "bright spot," where at each instance of time we can detect the centroid of the target (see Figure 4). In our present implementation, the distance to the target is obtained artificially.

The pan platform and the camera form one module of the system which can operate in two different modes — *exploratory mode* (State 0 in Figure 5) and *tracking mode* (State 1 in Figure 5). The exploration strategy associated with the first mode pans the camera around until the target is detected. Once

the target is detected we switch to the tracking mode where the goal is to keep the target in the center of the field of view (FOV). The control strategy applied in this case is depicted in Figure 3 where

$$\dot{\omega} = K_v.(\dot{x}_{ref} - \dot{x}_m) + K_p.(x_{ref} - x_m)$$

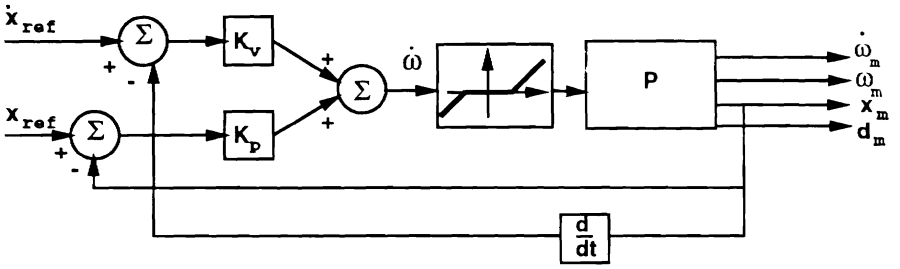



Figure 3: Pan control for target tracking: $d_m$ is the distance from the target, $\dot{\omega}$ is the turning velocity of the pan platform, $\omega_m$ and $\dot{\omega}_m$ are the current position and velocity estimates of the pan platform, $K_p$ and $K_v$ are position and velocity gain factors and $x_{ref}$ and $\dot{x}_{ref}$ are desired relative position and velocity which are 0

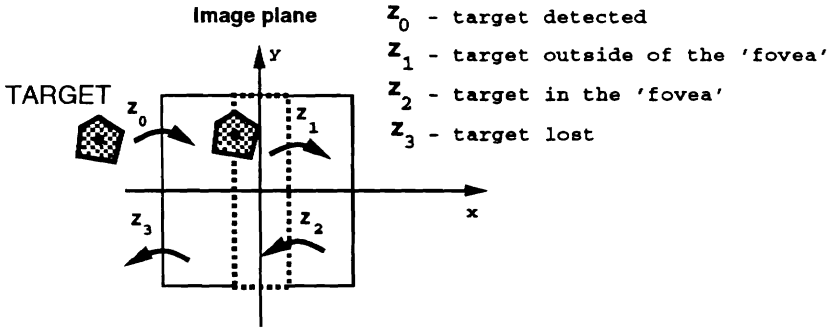



Figure 4: Tracking Events. Fovea is referred to the region in the center of the FOV throughout our examples.

In the case when the target is lost (the measurement $x_m$ exceeds the bounds in order to be used for the control law in Figure 3) we switch back

to the *exploration mode*. The state space of the tracking behavior can be partitioned into three regions: when the target is outside of the FOV, in the middle of the FOV (with some tolerance) and outside the center of the FOV (see Figure 4).

Change of control strategies is then driven by events, which correspond to the crossings of the boundaries between different regions of the state space. Note also that the direction of boundary crossing is important, introducing therefore some temporal (dynamic) aspect to the DES description. (e.g. event $z_0$ and $z_3$ are different). The finite state machine describing the behavior of this module is in Figure 5. State 0 corresponds to the *exploration mode* and State 1 to the *tracking mode* with the control strategy in Figure 3.

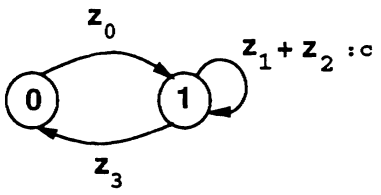



Figure 5: DES model for tracking

## 4.2 Mobile base

The mobile base has rotational and translational degrees of freedom and operates in two basic modes: *point-to-point* mode and *go* mode. The *point-to-point* mode uses trapezoidal velocity control profile to perform turns and straight line moves of a specified distance. The *go* mode moves the Labmate in a straight line at the current velocity setting. In this mode a continuous turn rate $\dot{\theta}$ can be superimposed on the existing forward velocity. The state of the system is fully determined by $(xpos, ypos, \theta, vel, tvel, mode)$, where $(xpos, ypos, \theta)$ is the current position and heading of the mobile base, and $vel$ and $tvel$ are its current linear and turning velocity settings. In both the

mobile base and pan platform the point-to-point mode uses position encoders (odometry) and corresponds to a simple *feedforward* control strategy, while motion in continuous mode corresponds to a *feedback* control strategy servoing on an external measurement determined by perceptual processes. One example of such servoing is in the case of tracking behavior where the mobile base servos on the "neck of the system" (pan platform) $\omega_m$ and the distance of the target $d_m$. The description of different modes (control strategies) in which the platform can operate follows.

The goal of the control strategy while tracking a target is to try to align with the neck of the system and at the same time try to keep the distance to the target constant. This is accomplished by the following control rule:

$$\begin{aligned} \dot{\theta} &= K_v.(\dot{\omega}_{ref} - \dot{\omega}_m) + K_p.(\omega_{ref} - \omega_m) \\ v &= K_d.(d_{ref} - d_m) \end{aligned}$$

where the block diagram is in Figure 6.

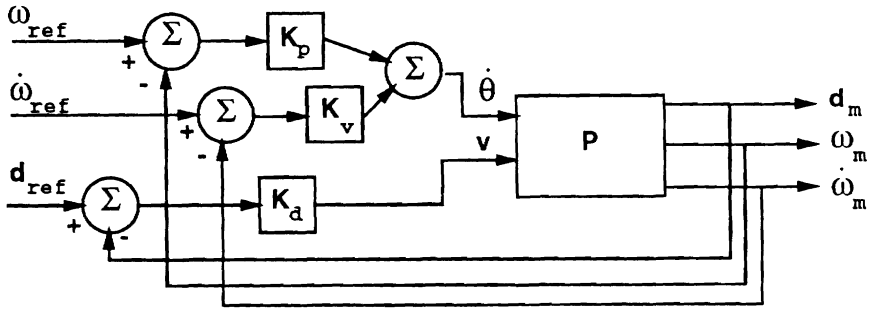



Figure 6: Servoing the neck: $\omega$ and $\dot{\omega}$ are current position and velocity of the pan platform, $v$ and $\theta$ are the linear velocity and turning rate of the base, $d_m$ is the current estimate of the target velocity

Within this mode we can distinguish two types of events which correspond to the crossing of the boundary of the desired operating region (see Figure 7).

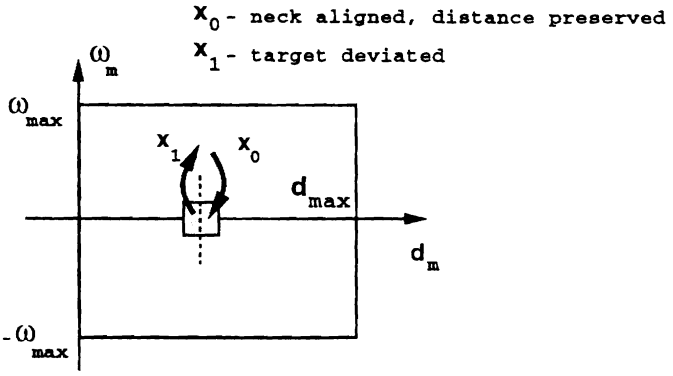



Figure 7: Events for neck servoing: $\omega_{max}$ is the maximal orientation of the pan platform with respect to the mobile base, $d_{max}$ is the maximal distance of the target

So far the transitions between states were caused by crossing the boundary of the state space of a particular module of the system. The control strategy however may also be changed due to the external asynchronous events generated by other participating processes. In our case there is a human operator which is a part of the system and can at any instant direct the platform to follow a given path. This would correspond to an event that can bring the mobile base module to another mode — "path following". In State 0 the mobile base is servoing on the neck of the system (control diagram in Figure 6) and in State 1 the base is following a given path. Event $x_2$ represents an asynchronous interrupt from the human operator (or the path planner) that the path is computed. Event $x_3$ is an attempt to correct for the deviation from an intermediate goal of the plan, and events $x_4$ and $x_5$ are asserted upon path completion or interruption.

## 4.3 Obstacle detection and avoidance

The third module of our system is the obstacle avoidance module. Obstacles are detected through the difference between a pair of stereo images after

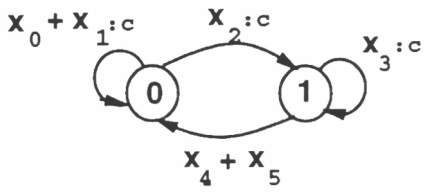



Figure 8: DES model of the robot process. Event $x_2$ is asserted when the path is computed, events $x_4$ and $x_5$ correspond to the path completion or interruption respectively, events $x_0$ and $x_1$ relate to the neck servoing control strategy.

Figure 9: a) Left Image; b) Map of the free space in lower resolution (obstacles in white); c) Right Image

applying the proper inverse perspective mapping proposed by [MBLB91]. Differences in perspective between left and right views are used to determine the presence of an obstacle and its approximate location. The subsequently computed map of the free space in the common field of view of both cameras [KB93] is used for obstacle avoidance maneuvers (see Figure 9). We chose to accomplish the avoidance maneuver in a purely reactive way. Based on the distance *dist* to the obstacle and the clearance *clear* by which we want to avoid the closest obstacle (see Figure 10) in the vehicle's path, we compute the appropriate turning rate $\dot{\theta}$ in the following way:

$$\gamma = atan(\frac{-clear}{dist}) \quad (1)$$

$$\dot{\theta} = K_t . \frac{\gamma . vel}{dist - safety} \quad (2)$$

The *safety* term parameterizes how close we want to come to the obstacle.

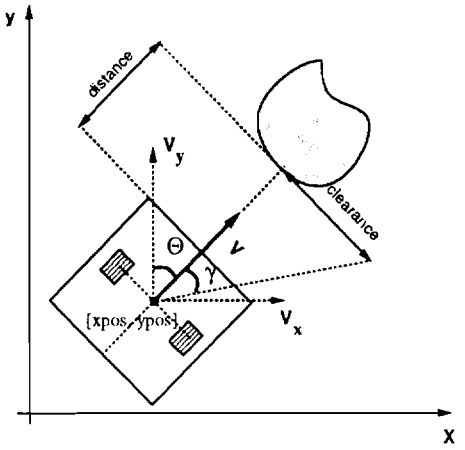



Figure 10: Steering away maneuver. $(x_{pos}, y_{pos}, \theta)$ is the current position and absolute heading of the mobile base in the world coordinate system. $\gamma$ is the angle by which the heading needs to be changed in order to avoid the obstacle.

The initial strategy of the obstacle avoidance process is just to monitor the free space ahead (State 0). When an obstacle is encountered the module switches the state and starts applying reactive *steer away* control strategy (State 1) until the path is again free (event $d_3$), or the obstacle becomes too close to steer safely away from (event $d_2$). Event partitioning for obstacle avoidance behavior is in Figure 11.

The choice of particular DES models as well as semantics of events is not unique. For example, in the obstacle avoidance process an event $d_0$ *obstacle detected* which triggers the avoidance control strategy can be asserted when the obstacle is 3m, 4m, etc. away. One can then think about the boundaries between different regions of the state space "sliding," where the position of the boundary is driven by some risk/cost functions determined by the task.

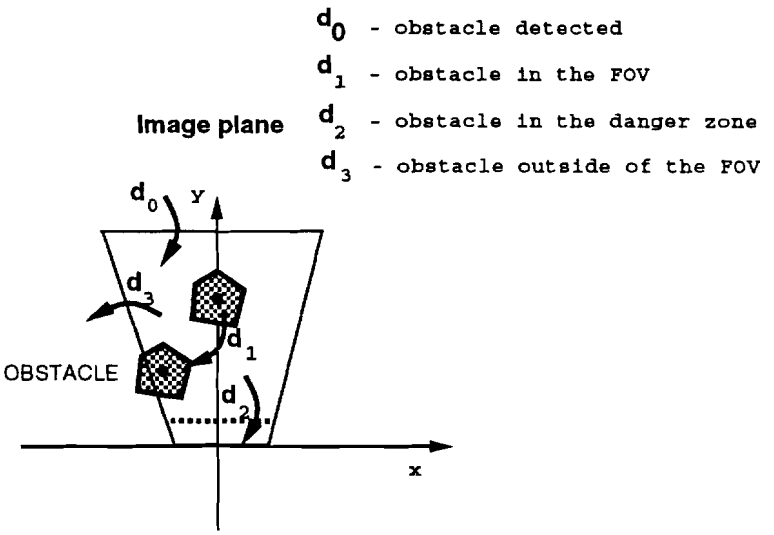



Figure 11: Events for obstacle avoidance. $x$ and $y$ coordinates span the common field of view of the stereo pair

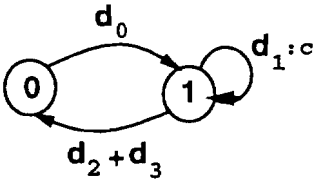



Figure 12: DES model of the obstacle avoidance process

## 4.4 Composite behaviors

The activation of different behaviors is closely related to the task to be accomplished. Composite behaviors are a combination of the elementary behaviors described in the previous section. Elementary behaviors in our case are controllable (in the discrete event sense) because either there is only one possible controllable event which can take place in each state (e.g. $d_1$ in the obstacle avoidance process), or the event which is suggested by the plant and enabled in a particular state is determined by the task and/or control strategy for that state (e.g. event $x_2$ path_computed). However, this might not be the case

for the combination of more than one behavior. Such situations can occur when the task of the agent is to follow a given target. This task requires an activation of "motor" behavior $P_1$ (Figure 8), target detection/tracking $P_2$ (Figure 5), and obstacle detection/avoidance behavior $P_3$ (Figure 12). Parallel execution of these behaviors may lead to possible conflicts. For example we can envision a situation when the tracking process is in State 1, following the target when the obstacle is detected. Suddenly the tracking behavior which assures that the mobile base is aligned with the camera pan platform has a conflicting goal with the obstacle avoidance process. To prevent this type of unwanted interaction some supervisory control is needed. Parallel execution of the component behaviors is modelled in an interleaving fashion, so the resulting behavior represents all possible sequences of events. The composition of $n$ behaviors $P_1, ..., P_n$ results in a new behavior, $P$, which is obtained as a **synchronous product** [RW87]

$$P = P_1 \| P_2 \| ... \| P_n$$

of the component behaviors. Constraints on the composite behaviors of the system can be expressed in terms of another finite state machine (see Figure 13. In our case the constraint is very simple — just expressing the fact that once the obstacle is detected the obstacle avoidance behavior takes over the control of the mobile platform until one of the events *clear_path* or *stopped* is asserted. By applying this constraint to the composite behavior of the system we can synthesize the resulting supervisor, guaranteeing satisfaction of the constraint. For this particular scenario, with four component processes the resulting supervisor has 8 states and 48 transitions but only four of these states have associated the control pattern disabling the events controlling the mobile base (i.e. $x_0$, $x_1$, $x_3$). The resulting behavior obtained by applying this 'discrete event' control strategy is that in the absence of obstacles the mobile base is coupled with the pan platform, but while avoiding obstacles

this coupling is violated and the base steers away from the obstacle while the tracking system is still able to keep track of the target.

This selection process of different paths subject to a risk/cost function can also be described as *cooperation between processes.* Thus, *cooperative behavior* in this context denotes the establishment and management of relationships between perception-action pairs. Consider the following example of cooperation between a path-following process and an obstacle-avoidance process. We select the path-following process as the nominal process which is essentially risk free when obstacles are not present. We define the nominal cost to be the path-traversal time. We define the risk cost to be $1/dist_{min}$, where $dist_{min}$ denotes the distance of closest approach to the obstacle. We assume that the endpoint $p_f$ of the nominal path is obstacle free. Thus, the decision problem P1 becomes: Minimize the path-traversal time, from the vehicle's present position to $p_f$, subject to the constraint that the vehicle maintains a minimum distance of closest approach $dist_0$ to the obstacle.

Similarly we have cooperation between obstacle avoidance and path following behavior, where upon observing the event 'obstacle detected' the obstacle avoidance behavior steers the mobile base away from the obstacle until the path ahead is again clear. When the system asserts a free-path, the vehicle switches behavior and follows a recomputed obstacle-free path that minimizes the traversal time to reach $p_f$ from its current position.

In both previous examples of cooperative behaviors the switch between different behaviors occurs upon observing a certain event. (This switching process can be also driven by some additional measures, which are in essence going to effect the assertion of an event.) We may ask why do we need supervisory control in these seemingly simple cases? Why not use just a simple interrupt routine? We hope that the reader will see that this methodology allows us to scale up easily to situations which need some additional (possibly

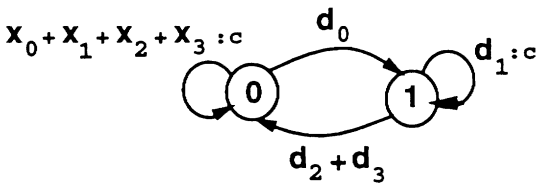



Figure 13: Composition constraint

task dependent) constraints. For example, when the agent is following a path we want to prevent gaze shifts or when the path is interrupted we want to reinvoke a path planner.

# 5 Conclusions and future work

In this paper we have posed the problem of systematic analysis and design of behaviors of artificial agents. To this end, a formalism of Discrete Event Systems has been selected as a suitable modelling tool for our purposes. In our investigations we have identified some primitive behaviors from which more complex behaviors can be composed, and have dealt with the issue of continuous control, as well as with discrete events and states. However, while the formalism of DES offers a systematic way of composing complex behaviors from primitive ones, additional constraints have to be added. The reason for this is that the visual sensors through which these agents perceive the world, and act upon it given the mobility task, add an extra degree of complexity to the system and necessitate the selection of an acquisition and processing strategy to obtain the qualitative and quantitative information needed to control the actuators of the system. The type of constraint we have proposed is a mutual exclusion constraint, which addresses the fact that different sensors generate commands to the same actuator.

In the case of purely visually guided navigation using a full camera head system this problem may not be present, due to the fact that there will be only

one resource for gathering visual information. In real situations, however, the mobile agent has several ways to accomplish its task, i.e. it must be able to select its path. In order to make this decision systematic, we have introduced the concepts of efficiency and safety of the agent. The efficiency is measured by the nominal cost (in our case, the traversal time) and the safety by the risk cost (in our case, the inverse of the distance between the agent and an obstacle).

The main advantage of employing the DES framework is that it enables us to synthesize the supervisor based on the task, including the cost functions. This methodology affords scaling up in the task space and environments. Thus far we have theoretically and experimentally investigated the control of composite behaviors within one agent and now are in the process of extending this to behavior of two, three and four agents navigating while keeping formation. The DES framework provides a transparent schema for the designer to analyze complex behaviors and hence guarantee the controllability of system which produces them.

# Acknowledgments

Navy Grant N00014-92-J-1647, AFOSR Grant 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770, and ASC 91 0813; Du Pont Corporation.

# References

[ALS93] P. J. Antsaklis, M. D. Lemmon, and J. A. Stiver. Learning to be autonomous: Intelligent supervisory control. Technical Report ISIS-93-003, Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 45656, 1993.

[Ark87] R. C. Arkin. Motor schema based navigation for a mobile robot. In *Proceedings Intl. Conf. on Robotics and Automation*, 1987.

[AS89] J. Aloimonos and D. Shulman. *Integration of Visual Modules.* Academic Press, Inc, 1989.

[Bal91] D. H. Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, February 1991.

[Bro86] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA - 2.(1):14 – 23, March 1986.

[BSM+89] B. Bhanu, P. Symosek, J. Ming, W. Burger, and H. Nasrand J. Kim. Qualitative target motion detection and tracking. In *DARPA Image Understanding Workshop*, 1989.

[CB91] D. J. Coombs and C. M. Brown. Cooperative gaze holding in binocular vision. *IEEE Control Systems Magazine*, 11(4):24–33, June 1991.

[FGS+92] M. Fossa, E. Grosso, G. Sandini, M. Zapendouski, F.Ferrari, and M. Magrassi. A visually guided mobile robot acting in indoor environments. In *Proceedings of IEEE Workshop on Applications of Computer Vision*, Palm Springs, CA, 1992.

[FHR+90] C. Fennema, A. Hanson, E. Riseman, J.R. Beveridge, and R. Kumar. Model-directed mobile robot navigation. Technical Report COINS TR 90-42, University of Massachusetts, Computer and Information Science, June 1990.

[HA91] X. Hu and N. Ahuja. Model-based motion estimation from long monocular image sequences. *University of Illinois at Urbana-Champaign*, 1991.

[Hor93] I. Horswill. A simple, cheap, and robust visual navigation system. In *Proceedings: From Animals to Animats II : Second International Conference on Simulation of Adaptive Behaviour.* MIT Press 1993, 1993.

[KB93] J. Kosecka and R. Bajcsy. Cooperation of visually guided behaviours. In *Proceedings ICCV 93*, Berlin, Germany, May 1993.

[MBLB91] H.A. Mallot, H.H. Bulthoff, J.J. Little, and S. Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological Cybernetics*, 64:177–185, 1991.

[RW87] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event processes. *SIAM J. Contr. Optimization*, 25(1):206-230, 1987.

[RW89] P. J. Ramadge and W. M. Wonham. The control of discerete event systems. *Proceedings of the IEEE*, 77(1):81–97, January 1989.

[Tea93] M. Tarr and M. Black et al. Panel discussion. In *Proceedings IJCAI'93*, pages 1661–1666, Chambery, France, August 1993. Morgan Kaufman.